

Requirements Definition

Getting to “What” and not “How”

2011 Montana Government IT
Conference

Deloitte. 



Agenda

1. The goal of requirements in systems development
2. Software development lifecycle considerations in requirements definition
3. The benefits of “What” and not “How” in requirements definition
4. Tips and tricks of generating “what” and not “how” requirements
5. Questions and Answers

The Goal of Requirements in Systems Development

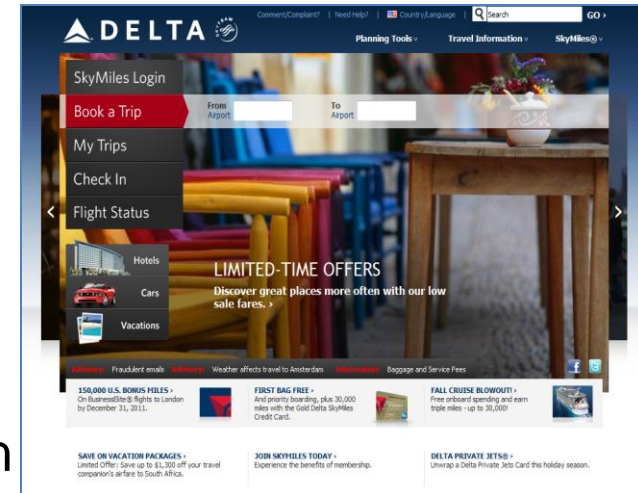
Why do we need requirements in systems development?

- A Systems Development project is similar to a physical construction project

| | Requirements (<u>What</u> is needed) | Design (<u>How</u> will it be realized) |
|-----------------------------|---|---|
| Construction Project | 3 bedrooms, 1.5 bathrooms, 2 car garage | Blueprints |
| Systems Development Project | Automated Eligibility Determination for SNAP and TANF | Detailed System Design Documents |

The Business Nature of System Development Projects

- Business System Examples
 - Airline booking
 - Auto insurance quotes and sales
 - Inventory management
 - Medicaid claims processing
 - Public assistance eligibility determination and issuance
- How are these different than software companies like Google or Microsoft?
- This drives the need for requirements

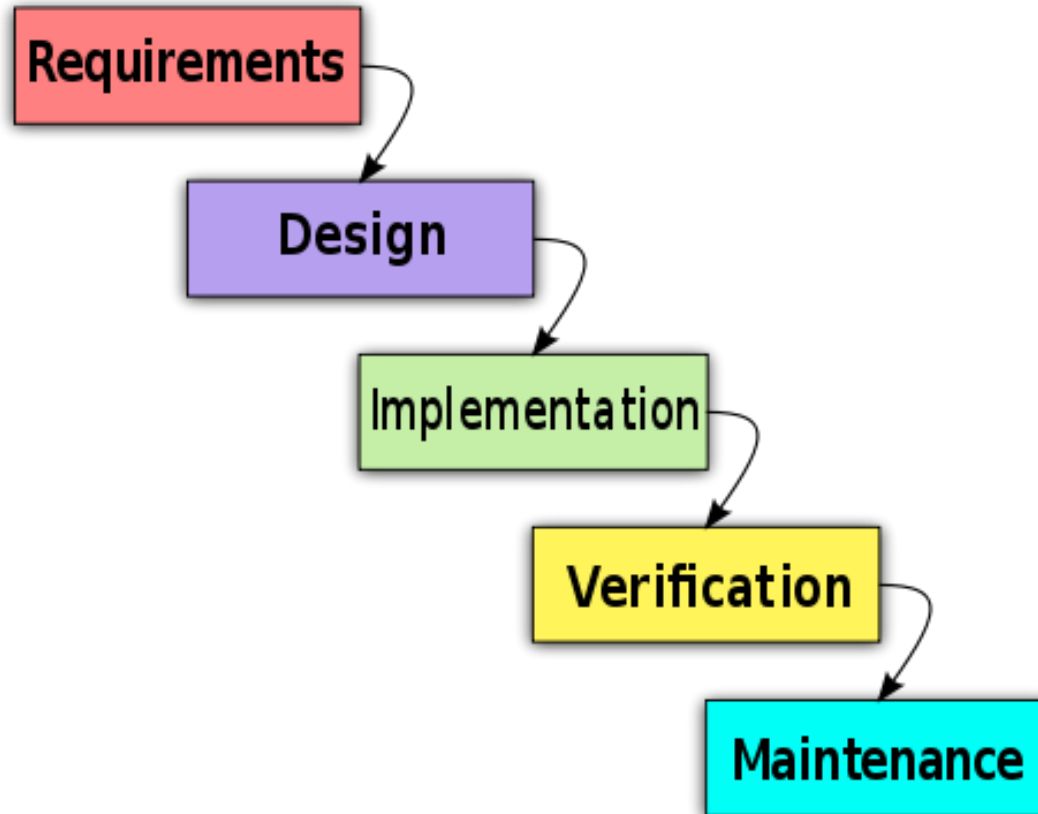


Software Development Lifecycle – Considerations with Requirements

- Overview of Common SDLC Methodologies
 - Waterfall
 - Iterative
 - Agile/Scrum
- Considerations to Requirements definition

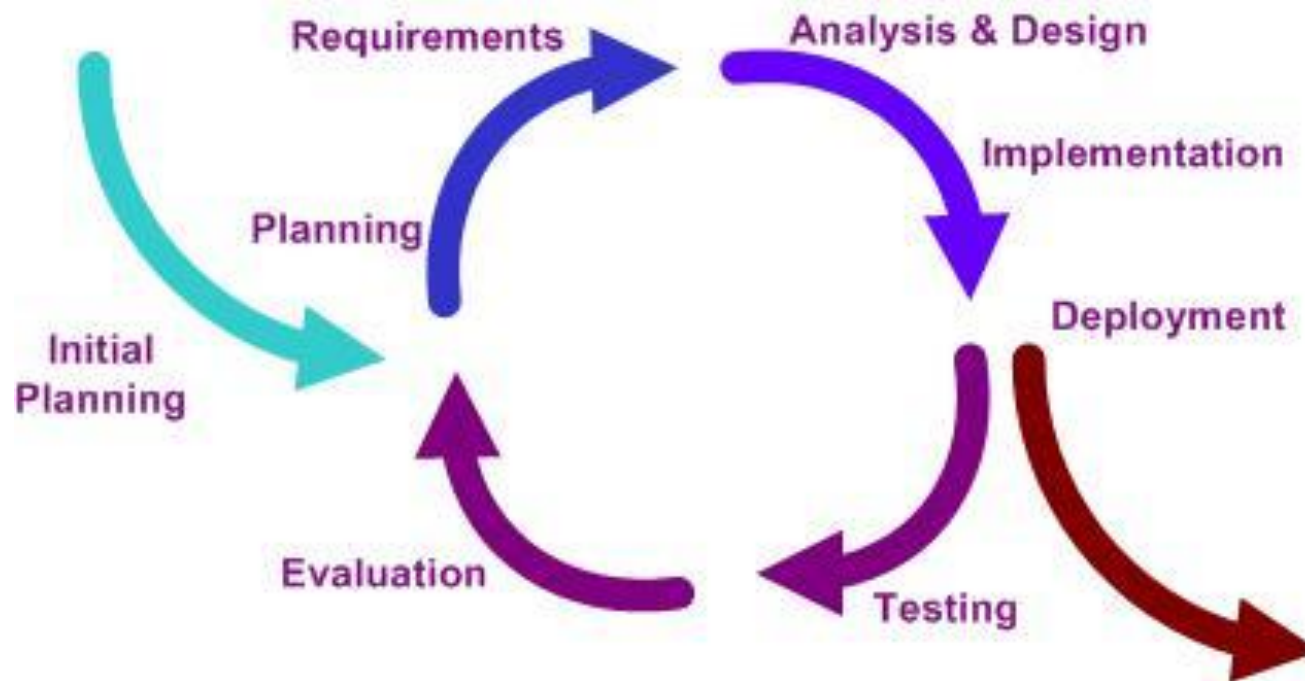
The Software Development Lifecycle – Overview of Methodologies

- Waterfall



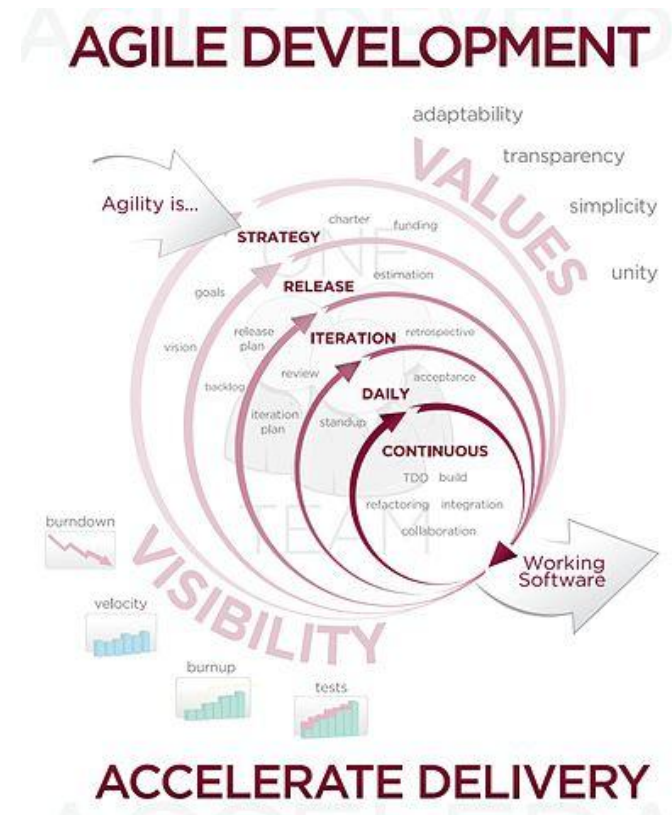
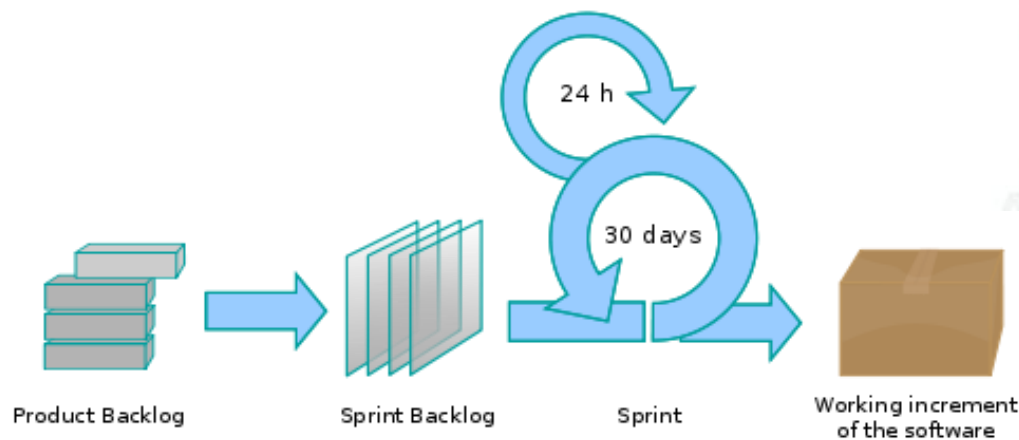
Overview of Methodologies - Continued

- Iterative

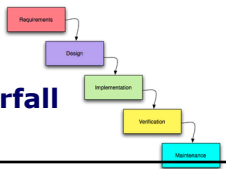


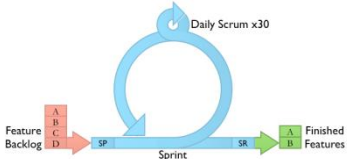


Overview of Methodologies - Continued

- Agile / Scrum



Overview of Methodologies – Requirements Commonality

| Methodology | Key Features / Uniqueness | Requirements Commonality |
|--|--|---|
| Waterfall  | <ul style="list-style-type: none"> - Closely mimics the phases of the Software Development Life Cycle (SDLC) - Requirements Gathering in a Waterfall Model has one-directional execution and needs to be completed in whole before design and build phases can proceed. | <ul style="list-style-type: none"> - There are subtle differences across methodologies ranging from # of iterations to level of detail, but most of them need to gather and document business requirements in some way, shape or form and pass them onto Build and Test teams for implementation and verification. This provides a significant opportunity for standardization across methodologies. |
| Iterative (RUP)  | <ul style="list-style-type: none"> - Follow an iterative approach for the entire SDLC. - Release-based methodology which enforces requirements to be captured in increments and every increment needs to complete it's SDLC before the next release/iteration commences. | <ul style="list-style-type: none"> - Methodologies like Agile are proactively promoting the adoption use cases as the primary form of requirements gathering approach. Having use case hierarchies with corresponding levels of detail complements Agile practices. |
| Agile  | <ul style="list-style-type: none"> - 3 to 4 iterations are typically recommended per release in Agile methodology. - Updates to requirements documents are frequent based on the end user feedback after an iteration has been built and deployed. | <ul style="list-style-type: none"> - Most of the differences across these methodologies are primarily around the functional requirements. However, System (including Non-Functional) requirements gathering can follow one consistent approach due to their static nature. |
| SCRUM  | <ul style="list-style-type: none"> - Follows a highly iterative model to complete a largely fixed set of prioritized backlog of requirements items in a series of short iterations called sprints. - From a requirements perspective, it promotes verbal communication across all team members and across all disciplines that are involved in the project A brief daily meeting (called a scrum), at which progress is explained, upcoming work is described, and obstacles are raised. | <ul style="list-style-type: none"> - Requirements Management is another aspect which is independent of methodologies used and needs to be addressed for both functional and non-functional requirements - Requirements Gathering Standardization across methodologies may be achieved either by aligning processes or leveraging tools or a combination of both. |

“What” versus “How” Requirements

- Examples from every day life of “What” versus “How”

| What | How |
|---|--|
| A ride to the airport at 1:00 PM | A ride to the airport at 1:00 PM in a black Cadillac Escalade driven by a man wearing a tuxedo |
| Clean the kitchen floor | Clean the kitchen floor using my toothbrush and a spray bottle of Windex. (Windex brand and not the store or generic brand) |
| Communicate the project status at least once weekly to the client management team | Communicate the project status at least once weekly to the client management team while wearing a fur coat and yelling the information |

Pitfalls of “How” requirements

- What was actually originally needed can get lost in the prescription of how to accomplish that need
- It can be challenging to verify that what was needed was accomplished
- The best possible approach may not be identified considering a bigger picture
- The requirements process may take longer yet yield less than optimal results

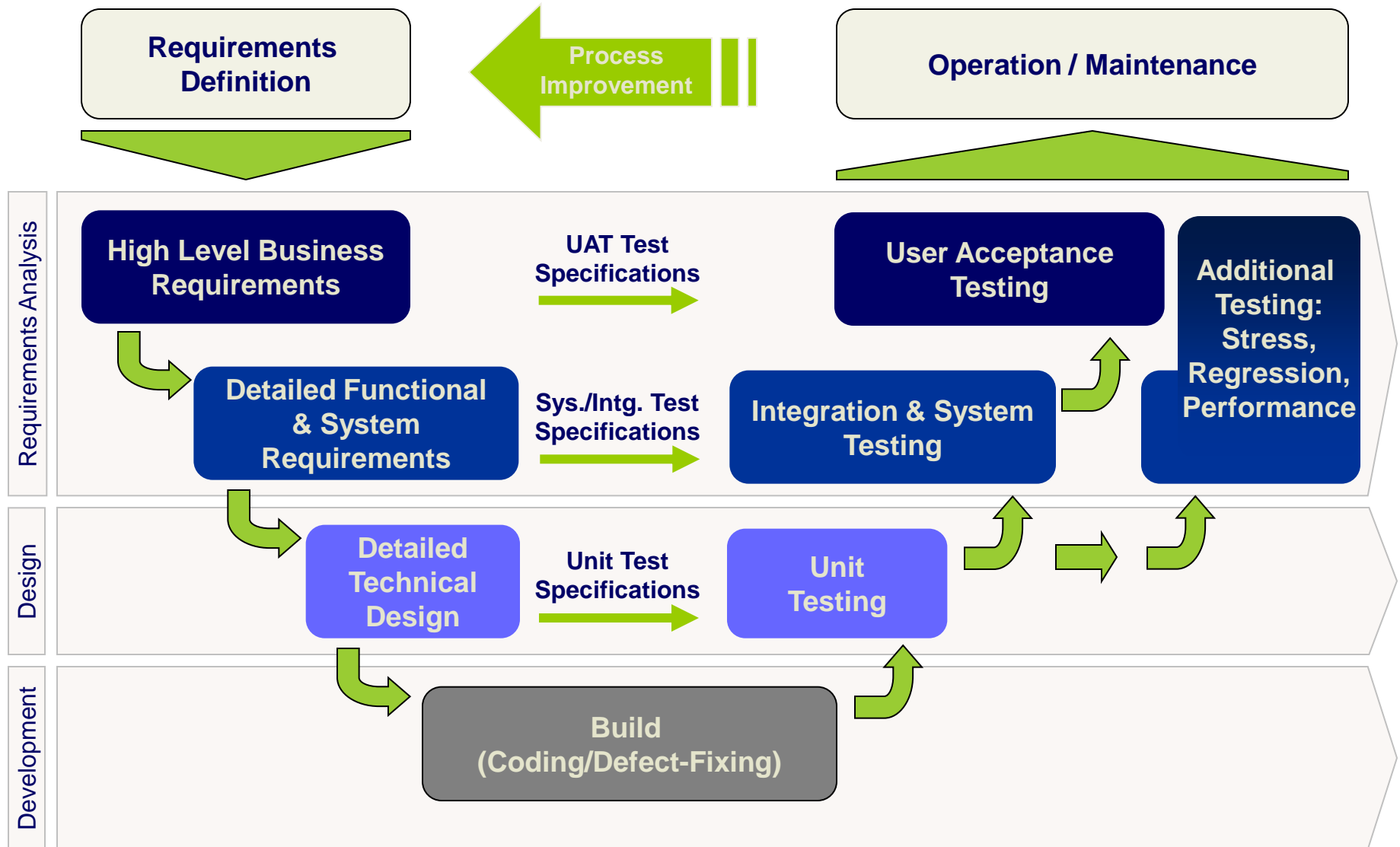


“How” Requirements – A Software Vendor’s Perspective

- Solution creativity may be limited with a prescribed approach
- Best practices may be neglected without proper consideration
- Ultimately, the overall solution may not be as economical or efficient to support and maintain
- Testing may lose a valuable input of the true “What “ requirements to ensure business needs are fulfilled



Properly defined Requirements Provide a Valuable Input to Testing



Tips and Tricks for Generating “What” and not “How” requirements

- Conducting Requirements sessions
- Documenting the Requirements
- Validating/Reviewing “What” not “How” requirements



Tips on Conducting Requirements Sessions

- Open with some examples of “What” not “How” requirements
- Provide relevant examples of “What” requirements translated to effective “How” designs
- Encourage participants to place their focus on defining “What” is the business need rather than how it will behave
- Tools:
 - Capture “How” ideas as design notes, however, not as the actual requirements
 - Explain the “parking lot” and know when to use it
 - Document action items well and follow up



Conducting Requirements Sessions – Typical ‘Rules of the Road’

- Be **physically** and **mentally present** (phone conference is not recommended)
- Please ... **Be on time** (to sessions, to individual meetings)
- Be **open**, **direct** and **focused** on the issues under discussion
- Listen first, **THEN** ask questions. **Respect** everyone’s input and feedback
- Be **constructive** and **creative** – discuss **ideas**, present an **alternative**
- **No lectures** please; be **concise**, be **respectful of everyone’s time**
- **Speak up** when you have a question or an alternative to offer, **otherwise quietly** signal your **agreement**
- **No laptops and mobile phones**, please (if you don’t need to pay attention, you probably don’t need to be there)
- Breaks are breaks
- Insist every person wear a **name tag** (indicating the representative’s business unit)
- “Because we have always done it that way” is **NOT** acceptable rationale for process and/or requirements.

Tips on Documenting Requirements

Focus on driving towards the ideal level of detail while writing the description of a requirement.



Too Broad

Without specifics and process context, level of effort cannot be estimated within a reasonable range

Example Requirement: "Notifications shall be based on a set of business defined criteria."



Ideal

Ideal requirements provide enough detail as well as context of a process

Example Requirement: "If an entitlement can not be verified for an SAS account, the system shall send a notification to both the Entitlement Group and the SAS Account Group."



Too Specific

Requirements that are too specific state an implementation approach rather than the requirement and/or does not include the necessary detail to implement

Example Requirement: "Ability to flag/pop-up a notification in red, bold text when entitlements can't be viewed for SAS."

Tips on Validating “What” not “How” Requirements - Continued

- Keep the following items in mind when documenting requirements
- Remember to include, not only the “What” but also:
 - Who/Whom
 - When
 - How Often
- Requirements should focus on “What” not “How”
 - E.g., describe what functionality is needed, not how it should work
- Remember that functionality may be delivered in more than one way
- Don’t lock onto a specific design by writing requirements that focus on the specifics of how a system should behave
- This is not a design session but a foundation for design
- Build requirements as a function of business process, not individual function points. Staying process-focused helps everyone understand how the system will function when deployed
- Properly consider channels and their differences when developing processes and gathering requirements





How to Contact US



Wil Carroll, Principal Deloitte Consulting

– wcarroll@deloitte.com



Kenny Smith, Senior Manager Deloitte Consulting, PMP

– kensmith@deloitte.com

- **Deloitte Consulting Helena Montana Office**
 - Suite 301, 350 N Last Chance Gulch, Helena, MT 59601



Deloitte.